WHAT IS CLAIMED IS:

1. A method for sharing a class among a plurality of applications in a multitasking computer system, the method comprising:

a first thread of a first application invoking a method of the class, wherein the method comprises an identifier, and wherein the identifier initially comprises an original value which associates the method with the first thread;

modifying the identifier in response to the first thread invoking the method such that the identifier comprises a temporary value, wherein the temporary value indicates that a single copy of the class is to be shared by the plurality of applications in the multitasking computer system;

exiting the method; and

modifying the identifier in response to the exiting the method such that the identifier comprises the original value which associates the method with the first thread.

2. The method of claim 1,

wherein the temporary value is a constant.

3. The method of claim 1,

wherein the class comprises one or more static variables, and wherein the temporary value indicates that a single copy of each of the static variables is to be shared by the plurality of applications in the multitasking computer system.

4. The method of claim 1, further comprising:

prior to the exiting the method, a second thread of a second application invoking the method of the class; and

the second thread accessing the single copy of the class.

5. The method of claim 1,
wherein the class is a system class.

6. The method of claim 1,
wherein the class is an application class.

7. The method of claim 1,
wherein the multitasking computer system comprises a virtual machine, and wherein the applications are executable by the virtual machine.

8. The method of claim 1, further comprising:
extracting one or more static fields from the class;
creating a separate copy of the one or more static fields for each of the plurality of applications that utilizes the class, wherein each of the separate copies corresponds to one of the plurality of applications; and
creating one or more access methods for the one or more static fields, wherein the access methods are operable to access the corresponding separate copy of the one or more static fields based upon the identity of the utilizing application.

9. The method of claim 1,
wherein the class comprises identifier-switching program instructions that are executable to perform the modifying the identifier in response to the invoking the method and the modifying the identifier in response to the exiting the method.

10. The method of claim 9,

wherein the class implements a protected interface using the identifier-switching program instructions, and wherein the identifier-switching program instructions are added to the class at load time.

11. The method of claim 1,

wherein the plurality of applications are executable in a platform-independent programming environment.

12. A system for sharing a class among a plurality of applications in a multitasking computer system, the system comprising:

a class comprising program instructions which are executable by a multitasking computer system; and

a plurality of applications which are executable by the multitasking computer system, wherein the plurality of applications comprises a first application, and wherein the first application comprises a first thread;

wherein the first thread of the first application is executable to invoke a method of the class, wherein the method comprises an identifier, and wherein the identifier initially comprises an original value which associates the method with the first thread; and

wherein the class is executable to:

modify the identifier in response to the first thread invoking the method such that the identifier comprises a temporary value, wherein the temporary value indicates that a single copy of the class is to be shared by the plurality of applications; and

modify the identifier in response to exiting the method such that the identifier comprises the original value which associates the method with the first thread.

13. The system of claim 12,

wherein the temporary value is a constant.

14. The system of claim 12,

wherein the class comprises one or more static variables, and wherein the temporary value indicates that a single copy of each of the static variables is to be shared by the plurality of applications.

15. The system of claim 12,

wherein the plurality of applications comprises a second application, wherein the second application comprises a second thread, and wherein the second thread is executable to:

invoke the method of the class; and

access the single copy of the class.

16. The system of claim 12,

wherein the class is a system class.

17. The system of claim 12,

wherein the class is an application class.

18. The system of claim 12, further comprising:

a virtual machine which is configured to execute the plurality of applications.

19. The system of claim 12,

wherein the class comprises identifier-switching program instructions that are executable to modify the identifier in response to invoking the method and modify the identifier in response to exiting the method.

20. The system of claim 19,

wherein the class implements a protected interface using the identifier-switching program instructions, and wherein the identifier-switching program instructions are added to the class at load time.

21. The system of claim 12,

wherein the plurality of applications are executable in a platform-independent programming environment.

22. A carrier medium comprising program instructions which are computer-executable to implement:

a first thread of a first application invoking a method of a class, wherein the method comprises an identifier, and wherein the identifier initially comprises an original value which associates the method with the first thread;

modifying the identifier in response to the first thread invoking the method such that the identifier comprises a temporary value, wherein the temporary value indicates that a single copy of the class is to be shared by a plurality of applications in a multitasking computer system;

exiting the method; and

modifying the identifier in response to the exiting the method such that the identifier comprises the original value which associates the method with the first thread.

23. The carrier medium of claim 22,

wherein the temporary value is a constant.

24. The carrier medium of claim 22,

wherein the class comprises one or more static variables, and wherein the temporary value indicates that a single copy of each of the static variables

is to be shared by the plurality of applications in the multitasking computer system.

25. The carrier medium of claim 22, wherein the program instructions are further computer-executable to implement:

prior to the exiting the method, a second thread of a second application invoking the method of the class; and

the second thread accessing the single copy of the class.

26. The carrier medium of claim 22,

wherein the class is a system class.

27. The carrier medium of claim 22,

wherein the class is an application class.

28. The carrier medium of claim 22,

wherein the multitasking computer system comprises a virtual machine, and wherein the applications are executable by the virtual machine.

29. The carrier medium of claim 22, wherein the program instructions are further computer-executable to implement:

extracting one or more static fields from the class;

creating a separate copy of the one or more static fields for each of the plurality of applications that utilizes the class, wherein each of the separate copies corresponds to one of the plurality of applications; and

creating one or more access methods for the one or more static fields, wherein the access methods are operable to access the corresponding separate copy of the one or more static fields based upon the identity of the utilizing application.

30. The carrier medium of claim 22,

wherein the class comprises identifier-switching program instructions that are executable to perform the modifying the identifier in response to the invoking the method and the modifying the identifier in response to the exiting the method.

31. The carrier medium of claim 30,

wherein the class implements a protected interface using the identifier-switching program instructions, and wherein the identifier-switching program instructions are added to the class at load time.

32. The carrier medium of claim 22,

wherein the plurality of applications are executable in a platform-independent programming environment.